Research

What is encryption?

- Encryption is the process of encoding the message whereas the decryption is the process of retrieving information from encoded message in decoded form.

- There are two types of cipher, one is a ***stream cipher*** and another is ***block cipher***.

- Cryptography is mainly of two types :

  1) ***Symmetric Algorithms*** : (“Secret key”) use the same key for both encryption and decryption
  2) ***Asymmetric Algorithms:*** (“Public key”) use different keys for encryption and decryption.

- Symmetric key approach has one major challenge may be disadvantage that is sharing the security with others asymmetric encryption answers this problem.

  - Using symmetric encryption, security requires that each pair of users share a secret key.

  - In an asymmetric system, each user has a public/private key pair.

- ***Symmetric-key cryptography is based on sharing secrecy.***

- ***Asymmetric-key cryptography is based on personal secrecy.***

Digital signature:-

Hash value of a message when encrypted with the private key of a person is his digital signature on that e-Document. They provide Authenticity, Integrity and Non-repudiation to electronic documents.

They provide authenticity by allowing one to verify whether the message he received is from the one whom he want to receive.

- Any message irrespective of its length can be compressed or abridged uniquely into a smaller length message called the Digest or the Hash.
- Smallest change in the message will change the Hash value

AES (Advanced Encryption Standard) :-

- It's the ***most widely used*** encryption Algorithm.

- It allows 3 types of keys : 1) 128 bit key (*10 Round Encryption Process*)
  2) 192 bit key (*12 Round Encryption Process*)
  3) 256 bit key (*14 Round Encryption Process*)

- Every round for encryption contains 4 steps : 1) Substitute Byte 2) Shift Row
  3) Column mixing 4) Addition of Round key
- As the number of rounds increases Encryption becomes more and more strong. It ***means AES 256 is the strongest encryption method among different AES methods***.

- ***AES algorithm is the Symmetric Encryption method.***

- For decryption, each round consists of the following four steps : 1) Inverse shift rows
  2) Inverse substitute bytes
  3) Add round key
  4) Inverse mix columns
  Here, in decryption process steps are same but order is different.

- The best thing about this algorithm is that ***every bit of cipher text depends on every bit of plain text.*** That doesn't mean that if you change a single bit of plaintext, algorithm changes every bit of cipher text.

- The last round for encryption does not involve the "Mix columns" step. The last round for decryption does not involve the "Inverse mix columns" step

- It takes 128 bit block of plain text and make a 4x4 matrix called "STATE ARRAY". Do the process on that matrix. And then takes new 128 bit block to process.

- Byte substitution step consists of using a 16 × 16 lookup table to find a replacement byte for a given byte in the input state array. The entries in the lookup table are created by using the notions of multiplicative inverses in GF (2^8) and bit scrambling to destroy the bit-level correlations inside each byte.

- The Shift Rows transformation consists of (i) not shifting the first row of the state array at all; (ii) circularly shifting the second row by one byte to the left; (iii) circularly shifting the third row by two bytes to the left; and (iv) circularly shifting the last row by three bytes to the left.

- In column mixing step each byte in a column is replaced by two times that byte, plus three times the next byte (in the same column) , plus the byte that comes next (in the same column), plus the byte that follows (in the same column). The words ‘next’ and ‘follow’ refer to bytes in the same column, and their meaning is circular.

The XOR Encryption

XOR is the acronym for "exclusive OR”. XOR encryption is based on the XOR logic. The following truth table shows the XOR logic:-

Input 1	Input 2	Output
1	1	0
1	0	1
0	1	1
0	0	0

The features of XOR encryption are:-

1. It is polyalphabetic encoding.
2. It is very hard to crack.
3. Uses a keyword to encrypt and also to decrypt.
4. Easy to encrypt if the person has the right key.

An example can be used to display the process:-

Let’s assign the alphabets some number (for instance assign number 0 to 31 to the 26 alphabets and 5 extra characters).

Let alphabet d be numbered 4.

Then its binary form would be 00100.

Let there be a key, a word, “PAST" whose first character is p.

Let its number be 15.

So it binary form would be 01111.

Now we use this key to encrypt the letter "d".

Letter d	:	0	0	1	0	0
Letter p	:	0	1	1	1	1
				|		
				XOR		
				|		
New number	:	0	1	0	1	1

This new number will correspond to a new letter.

In this way XOR encryption can be used for encoding.

For XOR decryption we follow the below mentioned steps:

Suppose this new pattern (number) 0 1 0 1 1 corresponds to a new letter, for instance, z. Now if this letter (z) is again XOR operated with the same letter (from the key, p in this case), we get back the letter, which was encrypted at first i.e. the letter d. This explained below:-

Letter z:	0	1	0	1	1
Letter p:	0	1	1	1	1
			|		
			XOR		
			|		
Letter d:	0	0	1	0	0

In this way XOR is easy to decrypt.

Base64

Base64 :

Base64 is a group of similar binary to text encoding scheme that represent binary data in an ASCII string format by translating it into a radix-64 representation.

Radix:

In mathematical numeral system, the radix or base is the number of unique digits including ZERO, used to represent numbers in a positional numeral system.

EXAMPLE:

The encoded value of **man** is **TWFu**.

ASCII of m = 77

a = 97

n = 110 which are 8 bit binary values 01001101,01100001 and 01101110.These three values are joined together into a 24-bit string, producing 010011010110000101101110.

Group of 6 bit (6 bits have maximum 2^6=64 different binary values) are converted into individual number from left to right

TEXT CONTENT	M	a	n
ASCII	77(0x4d)	97(0x61)	110(0x6e)
BIT PATTERN	0 1 0 0 1 1 0 1	0 1 1 0 0 0 0 1	0 1 1 0 1 1 1 0
INDEX	19	22	5	46
BASE64 ENCODED	T	W	F	u

As this example illustrated, Base64 encoding converts three octets into four encoded character.

When the number of bytes to encode is not divisible by 3(that is, if there is only one or two bytes of input for the last 24-bit block) then, Add extra bytes with value ZERO so there are 3 bytes.

If there was only one significant input byte, only the first two base 64 digits are picked (12 bit) & if there were two significant input bytes, the first three Base64 digits are picked (18 bits)'='character might be added to make the last block contain four Base64 character. As a result, when the last group contains one octet,the four significant bits of the final 6-bit

block are set to ZERO &when the last group contains two octets ,the two least significant bits of the final 6-bit block are set to ZERO.

Padding

The '==' sequence indicates that the last group contained only one byte, and '=' indicate that it contained two bytes.

EXAMPL :

```
Base64("")      = "    "              (no bytes used 0%3=0)
Base64("f")     = "Z9=="               (one byte used 1%3=1)
Base64("fo")    = "Zm8=0"              (two bytes used 2%3=2)
Base64("foo")   = "Zm9v"               (three bytes 3%3=0)
Base64("foob")  = "Zm9vYg=="           (four bytes 4%3=1)
Base64("fooba") = "Zm9vYmE="           (five bytes 5%3=2)
Base64("foobar") = "Zm9vYmFy"          (six bytes 6%3=0)
```

The ratio of output bytes to input bytes in 4:3.Given an input of n bytes, the output will be 4[n/3] bytes long, including padding character.

When decoding Base64 text, four characters are typically converted back to three bytes .A single '=' indicates that the four characters will decode to only two bytes, while '==' indicates that the four character will decode to single byte.

URL ENCODING

- WHAT IS URL ENCODING

  URL stands for uniform resource locator. This type of encoding is used to encode
  The characters that don't have standard ASCII values.
  The special character gets encoded into hexadecimal form.

- HOW IS IT ENCODED?

  The encoding starts with '%'.
  Then the hexadecimal form appears.

  For example:

  As an example, the hexadecimal number 2AF316 can be converted to an equivalent decimal representation. Observe that 2AF316 is equal to a sum of (200016 + A0016 + F016 + 316), by decomposing the numeral into a series of place value terms. Converting each term to decimal, one can further write:

  $(216 \times 163) + (A16 \times 162) + (F16 \times 161) + (316 \times 160) =$
  $(2 \times 4096) + (10 \times 256) + (15 \times 16) + (3 \times 1) = 10995.$

  Another example……
  "Hello Günter" in url encoded form will be : Hello+G%C3%BCnter.

OVERVIEW

We chose “data encryption” project for our ID module because of the following reasons :-

1. We enter the information, create accounts, share data and what not with this internet which is creeping into lives with pace, so this made us choose this project , so that we can get into the depth of this subject and contribute in this field .

2. It is a very fascinating topic and we knew that we will be able to research a lot in this field. The project can be used to encode and decode data files.

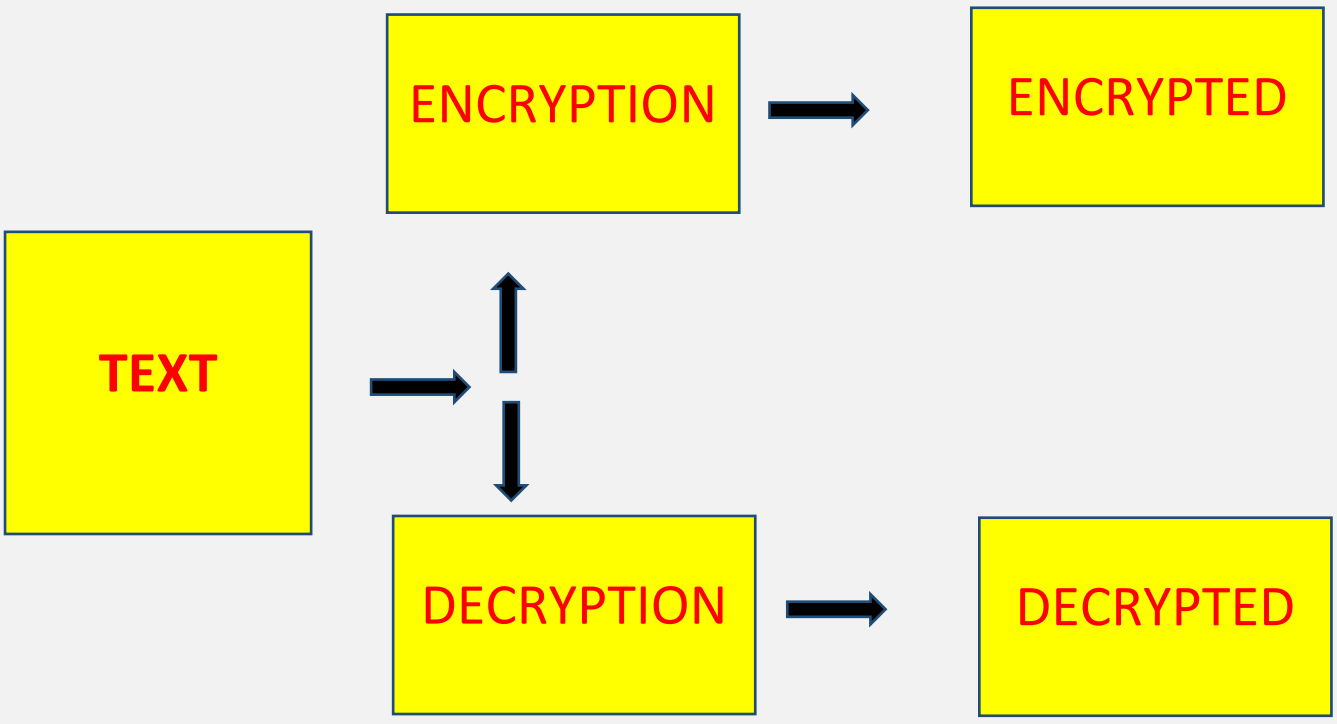
3. The project can let us know how the data is sent over networks and how middle man can access it , without our knowing . We also get to know how to secure this information from these unauthenticated users.

DESIGN

The design of the software is very user friendly. All the input and output activities will take place in terminal only. When the program will be executed the user will be asked if he wants to encode a text or decode an encrypted text.
Now if the user chooses to encode a text, he/she will be prompted to enter the text that the user wants to encrypt. After this the encrypted form will be displayed in the terminal itself and program gets terminated .

If the user chooses the other option i.e. decode, he will type in the encrypted text , get the decrypted text as an output in the terminal and the program will end.

This how the encryption program will work .



Bibliography

References

[1] Robert Sedgewick . *Algorithms* . Addison-Wesley Publishing Compay Inc.

[2]Wikipedia. *www.wikipedia.org*.

[3]Youtube. *www.youtube.com*

*[4]*Dr. Bill Young, Department of Computer Sciences, University of Texas *Foundations of computer security , Lecture 44 Symmetric and asymmetric encryption.*

THANK YOU !!